2021

4º Trabalho (Parte A)

Modelação e Programação

PROJECTO FINAL DE AVALIAÇÃO (PARTE-A)

DATA DE ENTREGA: 30/05/2022

Este trabalho tem por objetivo avaliar os conhecimentos que o aluno adquiriu em Modelação em Programação ao longo do semestre.

Importante: **Este trabalho é um exame individual** e como tal todo o código a ser avaliado terá que ser desenvolvido na íntegra pelo aluno. Se for comprovado que houve apoios externos à realização do trabalho este será imediatamente anulado e o aluno reprova à disciplina (inclusivamente o aluno com acesso à época especial). Estão excluídos, desta regra, os apoios pontuais que o docente possa dar na elaboração da arquitetura e na ideia geral do projeto a ser entregue na Parte A.

Planeamento e Desenho da Aplicação

Colocar no package tps.tp4 as classes da aplicação.

Este trabalho pretende consolidar e avaliar os conhecimentos adquiridos em Modelação e Programação dando ao aluno a oportunidade de mostrar a sua criatividade enquanto aprofunda os conhecimentos adquiridos ao longo do semestre. O objectivo do trabalho é desenvolvimento de uma aplicação usando a linguagem Java com interface gráfica (a ser introduzida posteriormente na Parte B deste trabalho). Nesse sentido, o aluno deverá aplicar os conhecimentos adquiridos na disciplina para criar um modelo de objetos para o domínio da aplicação a que se propõe desenvolver. O aluno deverá desenhar e defender perante o docente da disciplina os objetivos da aplicação e apresentar o diagrama de classes UML com todas as entidades, abstracções, interfaces e relações entre entidades e as acções/comportamentos aplicáveis às instâncias dos objetos dessas mesmas classes. A aplicação deverá também contemplar o armazenamento de dados (e.g. estado da aplicação) de forma persistente. Como tal, o aluno deverá também planear a gramática (DTD) e o exemplo de um documento XML que esteja válido de acordo com essa gramática.

O tema e as funcionalidades da aplicação é de escolha livre, mas sujeito a aprovação prévia por parte do avaliador. Os trabalhos podem ser algo como, por exemplo:

- Um jogo de tabuleiro como Damas, Xadrez, batalha naval, etc;
- Um jogo clássico como o Space Invaders, Tetris, Snake, Hive, Pac Man, Puzzle Bobble, Flappy bird, Arkanoid, Solitaire, Angry Birds, Mahjongg, Minesweeper, Sudoku, etc;
- Uma aplicação para gerir uma escola com várias turmas, disciplinas, alunos, professores e funcionários, etc.
- Uma aplicação para gerir um supermercado;
- Uma aplicação para gerir uma livraria;
- Uma aplicação genérica para ajudar coleccionadores a gerir as suas colecções.
- Uma aplicação para gerir um clube de atividades desportivas modalidades, sócios, etc;
- Uma aplicação simplificada para gerir uma linha ferroviária;
- Uma aplicação simplificada para gerir um aeródromo;
- Uma aplicação para gerir o controlo de acessos de um ginásio:
- etc.

A criatividade e a complexidade do trabalho realizado serão, obviamente, levadas em consideração na avaliação do trabalho. Na ausência de ideias, considerem uma das propostas do **Anexo I** e **II** como base de partida e desenvolva a partir dos requisitos aí descritos.

<u>Importante:</u> Trabalhos que não tenham sido pré-aprovados e avaliados positivamente até à data de entrega deste trabalho estão automaticamente reprovados.

Critérios e Regras de Avaliação

A nota deste trabalho (Parte A) tem um peso de 20 % da nota final da disciplina. Tal com no exame, para obter aprovação no trabalho (e consequentemente à disciplina) o aluno terá que obter na discussão o trabalho uma nota igual ou superior a 9.5 valores.

A nota final desta parte do trabalho resultará da seguinte ponderação: 75% nota da aplicação + 25% nota do relatório. Tanto a nota da aplicação como a nota do relatório terão que ser, individualmente, superiores a 9.5 valores.

Avaliação da Aplicação

O trabalho deverá abranger toda a matéria leccionada: classes, herança, interfaces, enumerados, tratamento de erros, coleccões, XML, DTD, XPATH, etc. Os alunos deverão também proactivamente procurar ir para além do que foi ensinado nas aulas para aprender e acrescentar riqueza ao trabalho. A complexidade da aplicação, a sua originalidade, a utilização de padrões de desenho, a qualidade da modelação, o domínio da matéria, a qualidade do código, a forma em como o código trata e recupera dos erros e a proactividade do aluno serão ponderados na avaliação da seguinte forma:

Originalidade do Projecto [Peso (0-1): 0.1]

Na avaliação premiamos ideias originais ou variantes originais de aplicações que já existam.

Complexidade [0.2]

A avaliação da complexidade é feita pelo docente tomando em consideração o esforço (em número de horas) que um aluno, que tenha feito os trabalhos práticos de laboratório, necessite para fazer o trabalho em apreciação.

Utilização e domínio de padrões de desenho [0.1]

A capacidade de autoaprendizagem e a utilização de padrões de desenho serão premiados na avaliação mediante a capacidade do aluno de mostrar conhecimentos da sua utilização. Para esse efeito, considere as seguintes referências:

https://www.researchgate.net/publication/307449818_GoF_Design_Patterns_with_examples_using_Java_and_UML
https://refactoring.guru/design-patterns/java

Qualidade da modelação [0.1]

Serão avaliados a estruturação da aplicação nas várias classes, propriedades dessas classes e relação entre classes. Será também avaliada a gramática DTD. As boas práticas da programação orientada a objectos deverão estar reflectidas nas escolhas feitas pelo aluno. A complexidade do trabalho está correlacionada também com o trabalho investido na modelação.

Tratamento e recuperação de erros (0.1)

A aplicação deverá utilizar o mecanismo de tratamento de excepções para gerir e recuperar os erros. Os alunos serão premiados nesta alínea se conseguirem integrar e utilizar a a biblioteca e API do Log4j: https://logging.apache.org/log4j/2.x/

Domínio da matéria (0.2)

O domínio da matéria, apesar de ter como ponderação 20% da nota da aplicação, é um factor de ponderação eliminatório. Isto é, se o aluno não conseguir demonstrar na discussão oral que domina a matéria do trabalho, ficará automaticamente reprovado.

Realização do código, testes realizados e qualidade da programação. (0.2)

A realização do código é um factor de ponderação eliminatório. Se o código associado às classes planeadas não estiver realizado e 100% funcional o aluno ficará automaticamente reprovado. O aluno deverá realizar, à semelhança do que foi feito na função *main()* dos trabalhos TP2 e TP3, um conjunto de testes e validações das várias classes.

Nota:

Embora o aluno possa utilizar componentes e APIs livremente disponíveis na Internet e a ideia da aplicação possa ter sido inspirada a partir de outra aplicação, o corpo principal do código da aplicação terá que ser obrigatoriamente desenvolvida de raiz pelo aluno sem apoio de terceiros, inclusivamente do professor.

Este trabalho equivale a um exame individual, como tal, pedir ajuda para fazer a aplicação equivale a copiar no exame e, como tal, resultará na anulação do trabalho.

Avaliação do Relatório

O relatório deverá estar bem estruturado com a introdução e os objectivos do trabalho. Deverá incluir uma apresentação clara dos objectivos do trabalho e uma breve discussão com citações de trabalhos relacionados. Deverá incluir os modelos UML de uma forma clara para o leitor. O relatório deverá incluir a descrição dos diagramas de classes, operações implementadas e outras informações que possam ser necessárias para que uma terceira pessoa entenda o contexto e consiga usar o trabalho como base para criar a sua própria aplicação.

O aluno como autor do relatório deverá tentar sempre, ao longo do processo de escrita do mesmo, posicionar-se no lugar do leitor e fazer uma avaliação do documento na sua principal função: documentar o trabalho realizado e transmitir informação e conhecimentos.

Sugestão: Utilize o latex para escrever o relatório. Se utilizar o latex poderá utilizar um vasto conjunto de packages para desenhar diagramas uml:

https://ctan.org/pkg/uml?lang=en)

https://www.overleaf.com/latex/templates/uml-diagrams-with-tikz-uml/ftfzzpmwnjqw)

e para listar código:

https://ctan.org/pkg/listings?lang=en

Nota: Não encha o relatório de código. Inclua apenas, se necessário, alguns trechos de código para descrever partes relevantes da aplicação.

O código deste trabalho e o relatório deverão ser entregues até ao dia **30 de maio de 2022**.

Bom trabalho, André Gomes, Carlos Júnior, João Ventura , Pedro Fazenda

Anexo I – Gestão de Contas Bancárias

O objectivo deste trabalho, em linhas gerais, é o desenvolvimento de uma aplicação JAVA para gestão de contas bancárias. O aluno deverá implementar um pacote de gestão de contas bancárias para informatizar as agências de um banco que, após o *login* de um utilizador com palavra-chave, tem as seguintes opções para manipular as contas bancárias:

1. Registo do utilizador

2. Autenticação

- 1 - Nome do utilizador
- 2 – palavra-chave

3. Sistema de gestão bancária

- 1 – Modificar data
- 2 – Saber data
- 3 – Criar conta
- 4 – Selecionar conta
- 5 – Saber o número da conta selecionada
- 6 – Depositar
- 7 – Levantar
- 8 – Saber saldo
- 9 – Saber total dos saldos das contas
- 10 – Terminar

O sistema suporta 4 tipos de contas: Conta Poupança-Habitação; Conta à ordem; Multibanco; Conta a prazo. Todas as contas têm um saldo. A conta a prazo tem a data do depósito. A conta Poupança-Habitação é uma conta a prazo cuja taxa de juro é de 10% nos primeiros 1000 euros e 5% no restante, mas só se pode levantar o dinheiro ao fim de 2 anos de existência da conta (a data tem de ser memorizada ao criar-se uma conta deste tipo). Nas contas à Ordem e Multibanco existe um registo de movimentos guardando-se para cada movimento o tipo (levantamento ou depósito nas duas contas e ainda consulta de saldos para a conta multibanco), a data e o valor. Quando se faz uma consulta de saldo numa conta à ordem ou multibanco é perguntado se o utilizador deseja listar o extrato do movimento.

4. Prémios

Existe um novo comando “Prémio”, que gera um número aleatório de conta existente. Se o número corresponder a uma conta à ordem, atribui 500 euros à conta por depósito. Se o número corresponder a uma conta a prazo, duplica o saldo corrente na conta. Se o número corresponder a uma conta poupança-habitação, incrementa em 0,1% a taxa de juro.

Anexo II – Gestão de uma Escola

O objectivo deste trabalho, em linhas gerais, é o desenvolvimento de uma aplicação JAVA para gestão de uma escola como o ISEL. O aluno deverá implementar um pacote de gestão de cursos, disciplinas, Semestres, Turmas, Alunos e Professores que, após o login de um utilizador com palavra-chave, tem as seguintes opções para manipular as contas bancárias:

1. Registo do utilizador

2. Autenticação

- 1 - Nome do utilizador
- 2 – palavra-chave

3. Sistema de gestão de inscrições

- 1 - Inscrever aluno numa disciplina e turma
- 2 - Listar alunos por turma
- 3 - Listar alunos por disciplina
- 4 - Ordenar alunos dor idade e média de curso
- 5 - Remover e Inserir disciplinas
- 6 - Remover e Inserir turmas
- 7 - Listar cursos, turmas, professores, alunos, etc.
- 9 - Inserir e remover professores
- 10 - Obter média das notas de uma disciplina
- 11 – Inserir e remover Semestres
- 12 - Terminar

O sistema permite gerir todo o ecossistema de classes que dão suporte às entidades supradescritas. Os alunos deverão ter as notas que obtiveram em todas as disciplinas e o registo histórico de todas as inscrições que foram feitas. As turmas estão limitadas a 30 alunos e cada disciplina tem um determinado números de créditos ECTS. Inspire-se na realidade do ISEL e no curso da LEIM, em particular, para realizar este trabalho.

4. Prémios

O sistema analisa as disciplinas concluídas por cada aluno e selecciona, para aqueles que nuca reprovara a nenhuma disciplina e que obtiveram a melhora nota do ano em que estão inscritos, os aluno que irão receber as bolsas de estudo por mérito.